Lucrarea nr. 3 - Folosirea argumentelor în linia de comandă, operații cu vectori

Breviar teoretic

Crearea unui vector

1. Declararea vectorului: `int[] sir;`
2. Instantierea - Se realizeaza prin intermediul operatorului `new` si are ca efect alocarea memoriei pentru vector, mai precis specificarea numarului maxim de elemente pe care îl va avea vectorul;

```
sir = new int[100];
```

3. Initializarea (optional) - Dupa declararea unui vector, acesta poate fi initializat, adica elementele sale pot primi niste valori initiale, evident daca este cazul pentru asa ceva. In acest caz instantierea lipseste, alocarea memoriei facându-se automat în functie de numarul de elemente cu care se initializeaza vectorul.

```
int []sir={77, 99, 44, 55,22,88,11,00, 66,33};

Ex:    int v[10];           //ilegal
       int v[]= new int[10];       //corect
```

Vectori multidimensionali

În Java tablourile multidimensionale sunt de fapt vectori de vectori.

```
Ex: int m[][];//declararea unei matrici
   m = new int[5][10];//cu 5 linii, 10 coloane
```

Obs: **`m[0], m[1], ..., m[5]`** sunt vectori de intregi cu 10 elemente

Dimensiunea unui vector

Cu ajutorul cuvântului cheie **`length`** se poate afla dimensiunea unui vector.

```
       int []x = new int[5];
       x.length are valoarea 5
       int m = new int[5][10];
       m[0].length are valoarea 10
```

Copierea vectorilor

Copierea unui vector în alt vector se face cu ajutorul metodei **`System.arraycopy`**:

```
       int x[] = {1, 2, 3, 4};
       int y[] = new int[4];
       System.arraycopy(x,0,y,0,x.length);
```

Vectori cu dimensiune variabila

Implementarea vectorilor cu numar variabil de elemente este oferita de clasa `Vector` din pachetul `java.util`. Un obiect de tip `Vector` contine numai elemente de tip `Object`.

Siruri de caractere

In Java, un sir de caractere poate fi reprezentat printr-un vector format din elemente de tip char, un obiect de tip String sau un obiect de tip StringBuffer.

Declararea unui sir

Daca un sir de caractere este constant atunci el va fi declarat de tipul String, altfel va fi declarat cu StringBuffer. (vezi "Clasele String, StringBuffer") Exemple echivalente de declarare a unui sir:

```
String str = "xyz";
char data[] = {'x', 'y', 'z'};
String str = new String(data);
String str = new String("xyz");
```

Concatenarea sirurilor

Concatenarea sirurilor de caractere se face prin intermediul operatorului +.

```
String str1 = "abc" + "xyz";
String str2 = "123";
String str3 = str1 + str2;
```

În Java, operatorul de concatenare + este extrem de flexibil în sensul ca permite concatenarea şirurilor cu obiecte de orice tip care au o reprezentare de tip şir de caractere.

```
System.out.print("Vectorul v are" + v.length + " elemente");
```

Folosirea argumentelor de la linia de comanda

O aplicație Java poate primi oricâte argumente de la linia de comanda în momentul lansarii ei. Aceste argumente sunt utile pentru a permite utilizatorului sa specifice diverse optiuni legate de functionarea aplicatiei sau sa furnizeze anumite date initiale programului.

Argumentele de la linia de comanda sunt introduse la lansarea unei aplicatii, fiind specificate dupa numele aplicatiei si separate prin spatiu.

Formatul general pentru lansarea unei aplicatii care primeste argumente de la linia de comanda este:

java NumeAplicatie [arg1 arg2 . . . argn]

În cazul în care sunt mai multe, argumentele trebuie separate prin spatii iar daca unul dintre argumente contine spatii, atunci el trebuie pus între ghilimele. Evident, o aplicatie poate sa nu primeasca nici un argument sau poate sa ignore argumentele primite de la linia de comanda.

In momentul lansarii unei aplicatii interpretorul parcurge linia de comanda cu care a fost lansata aplicatia si, în cazul în care exista, transmite aplicatiei argumentele specificate sub forma unui vector de siruri. Acesta este primit de aplicatie ca parametru al metodei main. Reamintim ca formatul metodei main din clasa principala este:

```
public static void main (String args[])
```

Vectorul primit ca parametru de metoda main va contine toate argumentele transmise programului de la linia de comanda.

De exemplu, sa presupunem ca aplicatia Sort ordoneaza lexicografic liniile unui fisier si primeste ca argument numele fisierului pe care sa îl sorteze. Pentru a ordona fisierul `"persoane.txt"` lansarea aplicatiei se va face astfel:

```
java Sort persoane.txt
```

In cazul apelului java Sort persoane.txt vectorul args va contine un singur element `args[0]="persoane.txt".`

Numarul argumentelor primite de un program este dat deci de dimensiunea vectorului `args` si acesta poate fi aflat prin intermediul atributului length al vectorilor:

```
numarArgumente = args.length;
```

Spre deosebire ce C/C++ vectorul primit de metoda main nu contine pe prima pozitie numele aplicatiei, întrucât în Java numele aplicatiei este numele clasei principale, adica a clasei în care se gaseste metoda main.

Exemplu: afisarea argumentelor primite la linia de comanda

```
public class Echo {
      public static void main (String[] args) {
      for (int i = 0; i < args.length; i++)
            System.out.println(args[i]);
      }
}
```

Un apel de genul `java Echo Drink Hot` Java va produce urmatorul rezultat:

```
Drink
Hot
Java
```

(aplicatia Echo a primit 3 argumente).

Una apel de genul java Echo "Drink Hot Java" va produce urmatorul rezultat:

```
Drink Hot Java
```

(aplicatia Echo a primit un singur argument).

Argumente numerice la linia de comanda

Argumentele de la linia de comanda sunt primite sub forma unui vector de siruri (obiecte de tip String). In cazul în care unele dintre acestea reprezinta valori numerice ele vor trebui convertite din siruri în numere. Acest lucru se realizeaza cu metode de tipul **parseXXX** aflate în clasa corespunzatoare tipului în care vrem sa facem conversia: Integer, Float, Double, etc.

Sa consideram, de exemplu, ca aplicatia Putere ridica un numar real la o putere întreaga, argumentele fiind trimise de la linia de comanda:

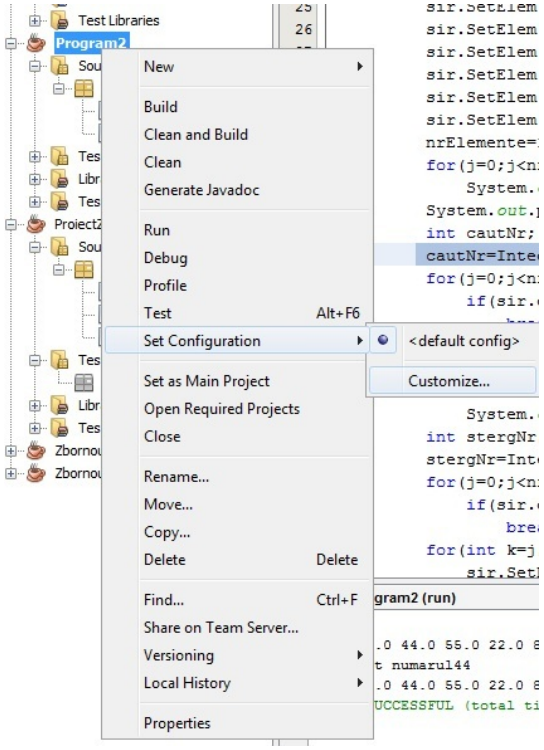
```
java Putere "2.1" "2"//ridica 2.1 la puterea 2
```

Conversia celor doua argumente în numere se va face astfel:

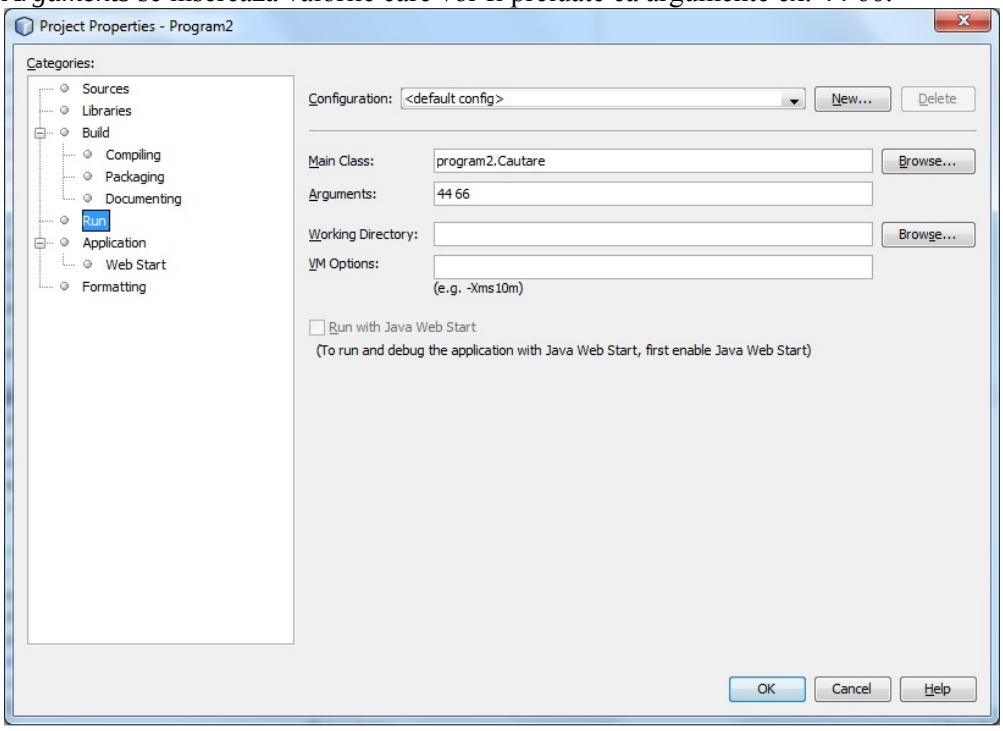
```
double numar;
int putere;
numar = Double.parseDouble(args[0]);
putere = Integer.parseInt(args[1]);
```

Metodele de tipul `parseXXX` pot produce exceptii (erori) de tipul NumberFormatException în cazul în care sirul primit ca parametru nu reprezinta un numar de tipul respectiv.

Inserarea argumentelor din Netbeans – se da click dreapta pe proiectul activ şi din meniul pop-up se alege opțiunea *Set Configuration -> Customize...* În câmpul *Arguments* se inserează valorile care vor fi preluate ca argumente ex. 44 66.



În câmpul *Arguments* se inserează valorile care vor fi preluate ca argumente ex. 44 66.



Constructori

La definirea unei clase, programatorul poate prevedea una sau mai multe metode speciale, numite **constructori**, al caror scop este acela de initializare a starii unui obiect imediat dupa crearea sa.
Caracteristicile unei metode **constructor**: are **acelasi nume** cu clasa in care este definita; un constructor nu returneaza nimic; va fi **apelata automat** la crearea unui obiect de tipul respectiv. Un constructor trebuie sa aiba modificatorul public in fata sa.
Daca programatorul nu prevede intr-o clasa **nici un constructor**, atunci compilatorul va prevedea clasa respectiva cu un **constructor implicit** fara nici un argument si al carui corp de instructiuni este vid. Daca programatorul include intr-o clasa cel putin un constructor compilatorul nu va mai genera constructorul implicit pentru acea clasa.

Creați un nou proiect Banca şi implementți următoarele două clase.

```
import java.io.*;

class ContBanca
{     private double balanta;
      public ContBanca(double balantainitiala)
      { balanta = balantainitiala; }

      public void depunere(double suma)
      { balanta = balanta + suma;
      System.out.println("A fost depusa suma de  "+suma);}

      public void retragere(double suma)
      { balanta = balanta - suma;
      System.out.println("A fost retrasa suma de  "+suma);}

      public void afiseaza()
      {System.out.println("balanta="+balanta);
      }
}

class OperatiiBanca

{ public static void main(String[] args)
      { ContBanca ba1 = new ContBanca(100.00);
System.out.print("Inainte de tranzactie, ");
      ba1.afiseaza();
      ba1.depunere(74.35);
      ba1.retragere(20.00);
      System.out.print("Dupa tranzactie ");
      ba1.afiseaza(); }
}
```

Creați un nou proiect VectorJava, unde se va declara un vector `int[] sir` şi întreg programul este conținut într-o singură clasă CautSir.

```
import java.io.*;
class CautSir
{ public static void main(String[] args) throws IOException
      {int[] sir;
      sir = new int[100];
      int nrElemente = 0;
      int j;
      int Nrcautare;
      sir[0] = 77;
      sir[1] = 99;
      sir[2] = 44;
      sir[3] = 55;
      sir[4] = 22;
      sir[5] = 88;
      sir[6] = 11;
      sir[7] = 00;
      sir[8] = 66;
      sir[9] = 33;
      nrElemente = 10;
      for(j=0; j<nrElemente; j++)
      System.out.print(sir[j] + " ");
      System.out.println("");
      Nrcautare = 66;
      for(j=0; j<nrElemente; j++)
      if(sir[j] == Nrcautare)
      break;
      if(j == nrElemente)
      System.out.println("Nu am gasit nici un numar  " + Nrcautare);
      else
      System.out.println("Am gasi numarul " + Nrcautare);

      Nrcautare = 55;
      for(j=0; j<nrElemente; j++)
      if(sir[j] == Nrcautare)
      break;
      for(int k=j; k<nrElemente; k++)
      sir[k] = sir[k+1];
      nrElemente--;

      for(j=0; j<nrElemente; j++)
      System.out.print( sir[j] + " ");
      System.out.println(""); }
}
```

Programul de mai jos este structurat pe clase, declararea vectorului este continuta in clasa DeclarSir iar numarul cautat in cadrul vectorului va fi trimis ca argument numeric in linia de comanda sau va fi setat ca parametru în câmpul *Arguments* în NetBeans .

```
import java.io.*;

class DeclarSir
{ private double[] a;

  public DeclarSir(int size) //constructor
     { a = new double[size]; }

  public void setElem(int index, double value)
     { a[index] = value;}

  public double getElem(int index)
     { return a[index];}
  }

class Cautare
{ public static void main(String[] args)
      { DeclarSir sir = new DeclarSir(100);
      int nrElemente = 0; int j;
      sir.setElem(0, 77);
      sir.setElem(1, 55);
      sir.setElem(2, 44);
      sir.setElem(3, 55);
      sir.setElem(4, 22);
      sir.setElem(5, 88);
      sir.setElem(6, 11);
      sir.setElem(7, 00);
      sir.setElem(8, 66);
      sir.setElem(9, 33);
      nrElemente = 10;

//parcurgerea si afisarea elementelor vectorului
      for(j=0; j<nrElemente; j++)
      System.out.print(sir.getElem(j) + " ");
      System.out.println("");

      int cautNr;
//numarul care va fi cautat este primit ca //argument numeric din linia de
comanda

      cautNr = Integer.parseInt(args[0]);
//aceasta valoare va trebui convertita din
//String in int prin folosirea functiei parseInt()

      for(j=0; j<nrElemente; j++)
      if(sir.getElem(j) == cautNr)
      break;

if(j == nrElemente)
System.out.println("Nu am gasit numarul" + cautNr);
else
System.out.println("Am gasit numarul " + cautNr);
```

```
        int stergNr;
        stergNr = Integer.parseInt(args[1]);
        for(j=0; j<nrElemente; j++)
        if(sir.getElem(j) == stergNr)
        break;
        for(int k=j; k<nrElemente; k++)
        sir.setElem(k, sir.getElem(k+1) );
        nrElemente--;

        for(j=0; j<nrElemente; j++)
        System.out.print( sir.getElem(j) + " ");
        System.out.println("");
        }
}
```

Probleme propuse:

1. Construiți în cadrul aplicației Banca clasa Client, în care introduceți atribute de tip nume, varsta, sex. Ataşați o serie de metode la această clasă internă şi realizați un nou program în al cărui program principal să folosiți clasa nou creată!
2. Construiți o altă clasă Banca, în care introduceți atribute de tip nume, adresa, capital. Ataşați o serie de metode la această clasă internă şi realizați un nou program în al cărui program principal să folosiți clasele nou create!
3. Creați o functie de stergere a unui element din cadrul unui vector - ex: public boolean sterge (int element) {}. Funcția va fi implementată în clasa DeclarSir.
4. Creați o functie pentru parcurgerea şi afisarea elementelor vectorului declarată în clasa DeclarSir.